Symbolic computation with finite biquandles

Conrad Creel
conradcreel@gmail.com

Sam Nelson
knots@esotericka.org

Department of Mathematics, University of California, Riverside
900 University Avenue, Riverside, CA, 92521

Abstract

A method of computing a basis for the second Yang-Baxter cohomology of a finite biquandle with coefficients in $\mathbb{Q}$ and $\mathbb{Z}_p$ from a matrix presentation of the finite biquandle is described. We also describe a method for computing the Yang-Baxter cocycle invariants of an oriented knot or link represented as a signed Gauss code. We provide a URL for our `Maple` implementations of these algorithms.

KEYWORDS: Virtual knots, finite biquandles, Yang-Baxter cohomology, symbolic computation
2000 MSC: 57M27, 57M25, 57-04

1 Introduction

In [7], algorithms for computing the values of quandle counting invariants on virtual knots and for finding the finite quandles which define these invariants were described. In this paper we describe algorithms for finding and computing a generalized form of this quandle counting invariant using a similar methodology – representing virtual knots and links as signed Gauss codes and representing finite biquandles as block matrices.

A *biquandle* is a non-associative algebraic structure defined on a set B whose axioms are motivated by thinking of the elements of B as *semi-arcs* in an oriented knot diagram and thinking of the four possible crossing relationships on inbound semi-arcs at a crossing as four binary operations on the set; the biquandle axioms are then the conditions required in order to preserve the algebraic structure under the three Reidemeister moves. Biquandles have been studied in recent papers such as [6], [10], [14] and more. The resulting algebraic structure is naturally a source of invariants of knots and links, both in the classical sense of disjoint unions of simple closed curves in S^3 and the combinatorial sense of Reidemeister equivalence classes of Gauss codes, also known as *abstract knots* or *virtual knots* [9], [8].

Perhaps the simplest easily computable link invariant derivable from a finite biquandle is the counting invariant $|\mathrm{Hom}(K,T)|$ which counts homomorphisms from the knot biquandle K into a finite target biquandle T. This counting invariant has been studied in the case of various finite biquandle structures defined algebraically (Alexander biquandles, quaternionic biquandles, etc.) in [6] and in the case of biquandles defined symbolically by biquandle matrices encoding the four operation tables in [14].

Simply counting homomorphisms, however, discards some information about the link by treating all homomorphisms as the same. One way to incorporate information about the link diagram in the set of biquandle homomorphisms is to use the homomorphism to define a *Boltzmann weight*, such as an integer power of a variable t, at each crossing; the product of these weights over all crossings defines an invariant of the colored knot diagram, and the sum of these weights over the set of all colorings defines a new knot invariant. The kinds of weight-assigning functions which make this construction work are elements of the second Yang-Baxter cohomology of the target biquandle as

defined in [2]; hence, to find these invariants we must compute the second cohomology of our target biquandle.

In this paper we describe an algorithm for computing the second Yang-Baxter cohomology of a finite biquandle with rational coefficients from its symbolic matrix presentation. We give some examples of symbolic computation of the resulting Yang-Baxter cocycle invariants for virtual links represented by Gauss codes.

The paper is organized as follows: In section 2, we define biquandles and describe ways of representing finite biquandles in `Maple`. In section 3 we describe a combinatorial generalization of knot theory known as virtual knot theory and describe how virtual knots and links may be represented in `Maple` as signed Gauss codes. In section 4 we describe our method for computing a basis for the second Yang-Baxter cohomology of a finite biquandle with coefficients in $\mathbb{Q}$ or a finite field $\mathbb{Z}_p$ and our method for computing the Yang-Baxter 2-cocycle invariants of a knot or link using these cocycles. Our `Maple` programs are available in the files `biquandles-maple` and `yangbaxtercohomology.txt` downloadable at `www.esotericka.org/quandles`.

2 Finite biquandles

We begin this section with the definition of a biquandle [11].

Definition 1 A *biquandle* is a set B with four binary operations $B \times B \to B$ denoted by

$$(a,b) \mapsto a^b,\ a^{\overline{b}},\ a_b, \quad \text{and} \quad a_{\overline{b}}$$

respectively, satisfying the following axioms:

1. For every pair of elements $a, b \in B$, we have

$$\text{(i) } a = a^{b\overline{b_a}}, \quad \text{(ii) } b = b_{a\overline{a^b}}, \quad \text{(iii) } a = a^{\overline{b}b\overline{a}}, \quad \text{and} \quad \text{(iv) } b = b_{\overline{a}a^{\overline{b}}}.$$

2. Given elements $a, b \in B$, there are elements $x, y \in B$, possibly but not necessarily distinct, such that

$$\text{(i) } x = a^{b\overline{x}}, \quad \text{(ii) } a = x^{\overline{b}}, \quad \text{(iii) } b = b_{\overline{x}a},$$
$$\text{(iv) } y = a^{\overline{b_y}}, \quad \text{(v) } a = y^b, \quad \text{and} \quad \text{(vi) } b = b_{y\overline{a}}.$$

3. For every triple $a, b, c \in B$ we have:

$$\text{(i) } a^{bc} = a^{c_b b^c}, \quad \text{(ii) } c_{ba} = c_{a^b b_a}, \quad \text{(iii) } (b_a)^{c_{a^b}} = (b^c)_{a^{c_b}},$$
$$\text{(iv) } a^{\overline{b}\overline{c}} = a^{\overline{c_{\overline{b}}}\overline{b^{\overline{c}}}}, \quad \text{(v) } c_{\overline{b}\overline{a}} = c_{\overline{a^b b_{\overline{a}}}}, \quad \text{and} \quad \text{(vi) } (b_{\overline{a}})^{\overline{c_{\overline{a^{\overline{b}}}}}} = (b^{\overline{c}})_{\overline{a^{c_{\overline{b}}}}}.$$

4. Given an element $a \in B$, there are elements $x, y \in B$, possibly but not necessarily distinct, such that

$$\text{(i) } x = a_x, \quad \text{(ii) } a = x^a, \quad \text{(iii) } y = a^{\overline{y}}, \quad \text{and} \quad \text{(iv) } a = y_{\overline{a}}.$$

If $B = \{x_1, x_2, \ldots, x_n\}$ is a finite biquandle, then we define the *biquandle matrix* of B to be the $2n \times 2n$ matrix

$$B = \left[\begin{array}{c|c} B_1 & B_2 \\ \hline B_3 & B_4 \end{array}\right]$$

where the i, j entry of the submatrix B_m is $(B_m)_{i,j} = k$ where

$$(x_k) = \begin{cases} (x_i)^{\overline{(x_j)}} & m = 1 \\ (x_i)^{(x_j)} & m = 2 \\ (x_i)_{\overline{(x_j)}} & m = 3 \\ (x_i)_{(x_j)} & m = 4. \end{cases}$$

To avoid confusing subscripts denoting elements of B with the biquandle operations, we generally drop the "x"s and just write $B = \{1, 2, \ldots, n\}$.

In `Maple`, we represent a finite biquandle as a list of four $n \times n$ matrices $[M[1], M[2], M[3], M[4]]$ where $M[i]$ is the matrix B_i for $i = 1, \ldots, 4$. Thus, the biquandle word $1^{2^{\overline{3_2}}}$ in our `Maple` notation is

$$M[2][1, M[4][2, M[3][3, 2]]].$$

If axioms (1)-(3) are satisfied, B is a *birack*. The notation can be simplified by denoting $S(a, b) = (b_a, a^b)$; then axiom (1) implies that $S : B \times B \to B \times B$ is invertible with inverse given by $S^{-1}(a, b) = (a^{\overline{b}}, b_{\overline{a}})$. This is known as *switch* notation; a map S is called a *switch map* if it satisfies the *set-theoretic Yang-Baxter equation*

$$(S \times \mathrm{Id})(\mathrm{Id} \times S)(S \times \mathrm{Id}) = (\mathrm{Id} \times S)(S \times \mathrm{Id})(\mathrm{Id} \times S).$$

A biquandle is then a set B with an invertible switch map $S : B \times B \to B \times B$ whose component maps additionally satisfy axioms (2) and (4). See [6] for more.

Example 1 The set $\mathbb{Z}_n = \{0, 1, 2, \ldots, n-1\}$ is a biquandle with operations

$$x^y = tz + (1 - st)y \mod n, \quad x_y = sx \mod n,$$

$$x^{\overline{y}} = t^{-1}z + (1 - s^{-1}t^{-1})y \mod n, \quad x_{\overline{y}} = s^{-1}x \mod n$$

where s, t are invertible elements of $\mathbb{Z}_n$. This is an example of an *Alexander biquandle*. See [10] and [13] for more.

In [14], an algorithm is described for finding all biquandle structures on a set with a given finite cardinality, as well as an algorithm for counting homomorphisms from a finitely presented biquandle described by a presentation matrix into the specified finite biquandle. The biquandle search algorithm works by taking a "partially completed" $2n \times 2n$ matrix with entries in $\{0, 1, 2, \ldots, n\}$ where a zero is considered a "blank;" the program then selects a zero entry and replaces the zero with nonzero entries, propagating the value through the matrix using the biquandle axioms and equations obtained from them and searching for contradictions. Zeroes are rated according to how many biquandle words in the list of axiom testing conditions will be completed if the zero is filled in, via a program called `ratezero`; the program selects a zero entry with maximal rating in order to move through the search space more efficiently. Any resulting matrices are appended to a working list, and the process repeats until all zeroes have been filled in.

Our `Maple` code includes a program `abq` which finds the biquandle matrix for the Alexander biquandle $\mathbb{Z}_n$ with a choice of s and t invertible elements of $\mathbb{Z}_n$, as a way of generating biquandle matrices with larger cardinalities.

3 Virtual knots

Knots and links (disjoint unions of simple closed curves in S^3) are usually represented as *knot diagrams*, which are 4-valent graphs embedded in an oriented surface Σ with vertices interpreted as crossings and decorated to indicate which strand passes over and which passes under.

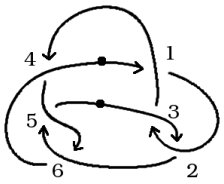
Various methods exist for encoding the information from a knot diagram in a more code-friendly way. One such method uses *Dowker-Thistlethwaite codes* or *DT codes*, e.g. the `knotscape`[1] package. For *oriented links* in which each component has a preferred choice of direction, we use signed Gauss codes to represent link diagrams.

A *signed Gauss code* for an oriented knot diagram K is an ordered list of crossing labels (including over/under information) and signs encountered as one travels the knot the direction of the

[1] Available at `http://www.math.utk.edu/~morwen/knotscape.html`

orientation. An oriented crossing is *positive* if, while looking in the positive direction of the overcrossing strand, the undercrossing strand is oriented right-to-left, and the crossing is *negative* if the under-strand is oriented left-to-right. If we have an oriented link with multiple components, we separate the Gauss code components with commas. For a given oriented link diagram, the corresponding signed Gauss code is well-defined up to ordering of the components and choice of base point for each component, with different choices of base point corresponding to cyclic permutations of the crossing labels. Given a signed Gauss code obtained from a link, we can reconstruct the original link diagram up to local isotopy, i.e., isotopy of the surface Σ in which the link diagram is embedded.

To represent an n-crossing Gauss code in `Maple`, we use a vector whose components are Gaussian integers $\pm X$ for a positive crossing and $\pm(X + \sqrt{-1})$ for a negative crossing where $X \in \{1, \ldots, n\}$, with the positive entry representing the overcrossing label and the negative the undercrossing label. We use a "0" entry as an "end of component" indicator to separate components in a link.

Example 2 The two-component link 

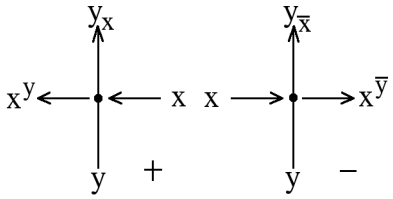
has signed Gauss code

$$[-1 - I, 2, -3, 1 + I, -4 - I, 5, -6, 4 + I, 0, 3, -2, 6, -5, 0]$$

in our `Maple` format.

Knots and links may be defined combinatorially as equivalence classes of link diagrams under the equivalence relation generated by the Reidemeister moves. If we restrict our attention to Gauss codes in which every crossing number appears once as an overcrossing and once as an undercrossing, with the same sign for both instances, then equivalence classes of such codes under the equivalence relation generated by the Gauss code Reidemeister moves are called *virtual knots*. These equivalence classes include some codes which do not correspond to knot diagrams which can be realized in the plane. Such Gauss codes may be interpreted as knot diagrams drawn on surfaces Σ with genus, corresponding geometrically to knots or links in thickened surfaces $\Sigma \times [0, 1]$ up to stabilization [4]. When such knots are drawn on genus-0 paper, the crossings which correspond to genus in the supporting surface, called *virtual crossings*, are shown as circled intersections to distinguish them from the ordinary *classical* crossings. In particular, virtual crossings have no over- or under-sense. Classical knots and links are then simply the subset of virtual knots and links which have representatives with supporting surface of genus 0. See [8] for more about virtual knots.

For any oriented virtual link L there is an associated biquandle, the *knot biquandle* $B(K)$, defined by assigning a generator for each *semi-arc* (the portion of a link diagram going from one over or undercrossing point to the next) and two relations at each crossing:



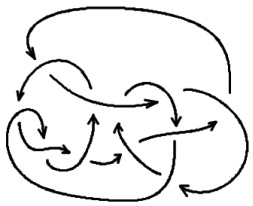
Thus, for any virtual knot or link K, we have an associated biquandle $B(K)$ defined via a universal algebra-style presentation with generators and relations; see [11] and [6] for more. If we label semi-arcs with generator numbers $1, 2, \ldots, n$ then such a presentation can be expressed as a biquandle presentation matrix with non-zero entries corresponding to relations coming from the crossings.

A homomorphism from the knot biquandle $B(K)$ to a finite target biquandle T can be regarded as a "coloring" of the knot diagram, i.e., an assignment of an element of T to every semiarc in the diagram such that above-pictured relations are satisfied at every crossing.

Initially, we used biquandle presentation matrices to represent knot biquandles for the purpose of counting homomorphisms into finite biquandles. The simplest method of computing $|\mathrm{Hom}(B(K), T)|$ is to generate a presentation matrix for $B(K)$ from a signed Gauss code, then test all possible maps from the set $\{1, 2, \ldots, 2n\}$ of generators of $B(K)$ to the target biquandle $T = \{1, 2, \ldots, m\}$ for the homomorphism condition (i.e., that the images of the generators of the knot biquandle satisfy the relations from the knot biquandle). However, for links with largish numbers of crossings this becomes very slow; an n-crossing link diagram has a knot biquandle with $2n$ generators, and brute-force checking all $(2n)^m$ maps where the target biquandle has cardinality m is impractical. An improved method is to use a "blank" homomorphism and propagate values through using the homomorphism condition in a manner analogous to our biquandle finding program. This is implemented in our `Maple` code as `bhomlist`.

An improved method of representing the knot biquandle of an oriented link is implemented in `bhomlist2` in the file `yangbaxtercohomology.txt`, available on `www.esotericka.org/quandles`. Rather than using a biquandle presentation matrix, the program `gauss2list2` takes a signed Gauss code and creates a list of biquandle relations using $M[1][i, j]$ for i^j, $M[2][i, j]$ for i_j, $M[3][i, j]$ for $i^{\overline{j}}$ and $M[4][i, j]$ for $i_{\overline{j}}$. The program `reducepreslist` then reduces the presentation by looking for generators which appear on one side of the relation but not the other; such generators are then eliminated, with all instances being replaced by the equivalent word in the remaining generators. This results in a dramatic reduction in the amount of brute-force checking required to compute $|\mathrm{Hom}(B(K), T)|$. Unfortunately, this method does require brute-force checking of maps; we cannot fill in and propagate values through since all surviving generators appear on both sides of any relation in which they appear. Nevertheless, this method is computationally much faster than the method of `bhomlist` since most knot biquandle presentations can be reduced significantly.

Example 3



This 11-crossing diagram of the Conway knot has knot biquandle with 22-generator presentation

$$\begin{aligned}\langle 1, 2, \ldots, 22 \quad | \quad & 1^{16} = 2,\ 2_{15} = 3,\ 3_8 = 4,\ 4^9 = 5,\ 5_{22} = 6,\ 6^{\overline{11}} = 7,\\ & 7_{\overline{20}} = 8,\ 8^3 = 9,\ 9_4 = 10,\ 10^{\overline{21}} = 11,\ 11_{\overline{6}} = 12,\\ & 12_{\overline{17}} = 13,\ 13^{\overline{18}} = 14,\ 14_{\overline{19}} = 15,\ 15^2 = 16,\ 16_1 = 17,\\ & 17^{\overline{12}} = 18,\ 18_{\overline{13}} = 19,\ 19^{\overline{14}} = 20,\ 20^{\overline{7}} = 21,\\ & 21_{\overline{10}} = 22,\ 22^5 = 1\rangle.\end{aligned}$$

`gauss2pres2` reduces this to a 5-generator presentation with generators $1, 8, 15, 16$, and 21 and relations too awkward to list here, though the interested reader can readily generate them with our code. For computing $|\mathrm{Hom}(K, T)|$ with $|T| = 4$, this reduces the number of brute-force checks from 22^4 to 5^4, an improvement of 37,481%; moreover, the number of relations which must be checked at each step is also reduced from 22 to 5.

Given a homomorphism obtained from the reduced biquandle presentation from `gauss2pres2`, we can recover the corresponding coloring of the diagram by using `bhomcomplete`. We will need these complete colorings for computing the Yang-Baxter cocycle invariant in the next section.

4 Yang-Baxter cocycle invariants

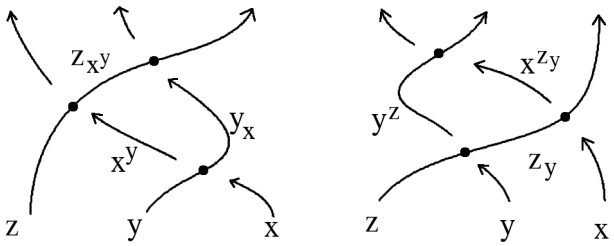
For any link L and finite biquandle T, the cardinality of the set of biquandle homomorphisms from the knot biquandle $B(L)$ to T is an invariant of link type. A set is more than a mere cardinality, however, and we'd like to try to extract more information about the link type from the set of biquandles colorings.

One way to do this, described in [2], is to use *Yang-Baxter cocycles*. The authors define a homology theory for finite biquandles with n-chains generated by ordered n-tuples of biquandle elements. The boundary map is defined using a bijection between ordered n-tuples in T^n and colorings of the n-cube graph Q_n with biquandle elements defined by interpreting an n-tuple as a coloring of the edges in a "preferred path" from the origin to $(1, 1, \ldots, 1)$ in the n-cube graph Q_n. Such a coloring extends to a unique *Yang-Baxter coloring* of the whole graph. The boundary map from standard cubical homology then induces via this bijection a boundary map $\partial_n^{YB} : A[T^n] \to A[T^{n-1}]$ with coefficients in an abelian group A. The resulting homology and cohomology theories are the *Yang-Baxter homology and cohomology* of the biquandle. We will primarily use $\mathbb{Q}$ as our coefficient ring for ease of computation; our software includes both $\mathbb{Q}$ and $\mathbb{Z}_p$ versions of our programs for p prime.

An alternative description of $H^2_{YB}(T; \mathbb{Q})$ in terms of knot diagrams makes it clear how Yang-Baxter 2-cocycles can be used to retain some information from the sets of biquandle colorings of a link diagram. Given a link diagram L with a coloring by elements of T, we will define a function $\phi : T \times T \to \mathbb{Q}$ such that the sum of the values of ϕ on the inbound crossings at each positive crossing and on the outbound crossings at each negative crossing, always with the color on the undercrossing semi-arc listed first, is unchanged by the Reidemeister moves. The value $\phi(x, y)$ at a crossing is called the *Boltzmann weight* of the crossing. Invariance under the second Reidemeister move is guaranteed by the convention just described; invariance under the third Reidemeister move requires

$$\phi(x, y) + \phi(x^y, z) + \phi(y_x, z_{x^y}) = \phi(x, z_y) + \phi(y, z) + \phi(x^{z_y}, y^z)$$

which is precisely the condition that $\phi \in H^2_{YB}(T; \mathbb{Q})$.



Invariance under the first Reidemeister move requires that $\phi(x, a) = 0$ and $\phi(a, y) = 0$, where x and y are the elements associated to a in biquandle axiom 4, for every $a \in T$. The subspace of $H^2_{YB}(T; \mathbb{Q})$ satisfying this condition is the *reduced Yang-Baxter cohomology* of T with coefficients in $\mathbb{Q}$. For each Yang-Baxter 2-cocycle $\phi : T \times T \to \mathbb{Q}$, the sum $\sum \phi(x, y)$ of Boltzmann weights over the set of all crossings in a biquandle-colored link diagram is unchanged by Reidemeister moves and can be understood as a kind of "signature" of the coloring homomorphism.

For such a cocycle ϕ, we compute the *Yang-Baxter cocycle invariant* of a link L by computing for each coloring $f \in \mathrm{Hom}(B(L), T)$ of L by T the sum over all crossings in L of the Boltzmann weights

$\phi(x,y)$ as above. Then, the set with multiplicities of these sums for all colorings of L by T is an invariant of links which contains more information than the cardinality of the set. For convenience, we can make each such sum the exponent of a formal variable t and then take the sum over the set of colorings; there is no loss of information here since we are adding only powers of t with coefficients of 1. Then,

$$\Phi_{YB}(L,T,\phi) = \sum_{f \in \mathrm{Hom}(B(L),T)} t^{\sum \phi(x,y)}.$$

Note that cohomologous cocycles define the same invariant.

To compute a basis for the rational cohomology vector space $H^2_{YB}(T;\mathbb{Q})$ from a biquandle matrix for T, we first make a matrix with rows corresponding to triples of biquandle elements and columns corresponding to characteristic functions $\chi_{(x,y)}(v,w) = \begin{cases} 1 & (v,w)=(x,y) \\ 0 & \text{else} \end{cases}$ on pairs of biquandle elements; for each triple (x,y,z) the entries in the columns corresponding to $\chi_{(x,y)}$, $\chi_{(x^y,z)}$ and $\chi_{(y_x,z_{x^y})}$ are set equal to 1, those corresponding to $\chi_{(x,z_y)}$, $\chi_{(y,z)}$ and $\chi_{(x^{z_y},y^z)}$ are set equal to -1 and all other entries are 0. $H^2_{YB}(T,\mathbb{Q})$ is then the null space of this matrix. To find a basis we use Gauss-Jordan elimination and then apply `getkernel` to read off a basis which is consistent between runs.

The program `ybcohom` then eliminates cohomologous cocycles by comparing the basis vectors from `ybcocycles` pairwise and keeping only one vector from each pair whose difference is a coboundary. Finally, `redybcohom` eliminates cocycles which do not satisfy the condition arising from the type I Reidemeister move.

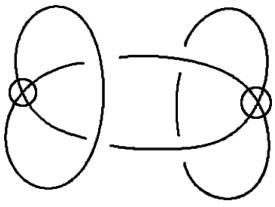
Our program `ybinv` takes as input a signed Gauss code, a biquandle matrix, and a Yang-Baxter 2-cocycle and computes the Yang-Baxter 2 cocycle invariant. Another program `ybinv2` takes a signed Gauss code and biquandle matrix and computes a basis for the reduced cohomology of T, and for each such cocycle computes the resulting cocycle invariant, outputting a vector of cocycle invariant values. Our use of `getkernel` ensures that the cocycles (and hence their invariants) are listed in the same order between runs.

Finally, we have modified versions of `ybinv` and associated programs, called `redybcohommodn`, `ybinvmodn`, etc. in which we replace $\mathbb{Q}$ with the finite field $\mathbb{Z}_p$ for p prime.

Example 4 The trivial cocycle $\phi(x,y) = 0$ has Yang-Baxter cocycle invariant $\Phi_{YB}(L,T,0) = |\mathrm{Hom}(B(L),T)|$ equal to the biquandle coloring invariant.

Example 5 If the finite target biquandle T satisfies $a_b = a_{\overline{b}} = a$ for all $a,b \in T$ then T is a *quandle* and the Yang-Baxter 2-cocycle invariants associated to T are the CJKLS state-sum invariants described in [3].

Example 6 This virtual knot is one of the *Kishino knots* introduced in [12]. The fact that it is not equivalent to the unknot, which was shown in [1] using quaternionic biquandles, shows that the operation of connected sum for virtual knots, unlike the classical case, is dependent on the portion of the knot in which the sum is performed. The biquandle

$K =$ 

$$T = \left[\begin{array}{cccc|cccc} 1&4&2&3&1&3&4&2 \\ 2&3&1&1&3&1&2&4 \\ 4&1&3&2&2&4&3&1 \\ 3&2&4&1&4&2&1&3 \\ \hline 1&3&4&2&1&4&2&3 \\ 3&1&2&4&2&3&1&4 \\ 2&4&3&1&4&1&3&2 \\ 4&2&1&3&3&2&1&4 \end{array}\right]$$

has reduced Yang-Baxter 2nd rational cohomology with basis $\{\phi_1,\phi_2\}$ where

$$\phi_1 = -\chi_{(1,3)} - \chi_{(2,1)} - \chi_{(2,3)} - \chi_{(3,2)},$$

$$\phi_2 = \chi_{(1,3)} + \chi_{(1,4)} + \chi_{(2,1)} - \chi_{(2,3)} + \chi_{(3,1)} - \chi_{(3,4)}.$$

Our program `ybinv` computes the Yang-Baxter 2-cocycle invariant values

$$\Phi_{YB}(K,T,\phi_1) = 12 + 2t^{-1} + 2t \quad \text{and} \quad \Phi_{YB}(K,T,\phi_1) = 12 + 2t^2 + 2t^{-2}.$$

We note that both of these invariants distinguish the Kishino knot K from the unknot, which has a value of $\Phi_{YB}(K,T,\phi_i) = 4$.

References

[1] S. Budden and R. Fenn. The equation $[B,(A-1)(A,B)] = 0$ and virtual knots and links. *Fundam. Math.* **184** (2004) 19-29.

[2] J. S. Carter, M. Elhamdadi, M. Saito. Homology theory for the set-theoretic Yang-Baxter equation and knot invariants from generalizations of quandles. *Fund. Math.* **184** (2004) 31-54.

[3] J. S. Carter, D. Jelsovsky, S. Kamada, L Langford and M. Saito. Quandle cohomology and state-sum invariants of knotted curves and surfaces. *Trans. Am. Math. Soc.* **355** (2003) 3947-3989.

[4] J. S. Carter, S. Kamada, M. Saito. Stable Equivalence of Knots on Surfaces and Virtual Knot Cobordisms, *J. Knot Theory Ramifications* **11** (2002) 311-322.

[5] C. H. Dowker and M. B. Thistlethwaite. On the classification of knots. *C. R. Math. Acad. Sci., Soc. R. Can.* **4** (1982) 129-131.

[6] R. Fenn, M. Jordan-Santana, and L. Kauffman. Biquandles and virtual links. *Topology Appl.* **145** (2004) 157-175.

[7] R. Henderson, T. Macedo and S. Nelson. Symbolic Computation with finite quandles. *J. Symbolic Comput.* **41** (2006) 811-817.

[8] L. H. Kauffman. Virtual knot theory. *European J. Combin.* **20** (1999) 663-690.

[9] N. Kamada and S. Kamada. Abstract link diagrams and virtual knots. *J. Knot Theory Ramifications* **9** (2000) 93-106.

[10] L. H. Kauffman and V. O. Manturov. Virtual biquandles. *Fund. Math.* **188** (2005) 103-146.

[11] L. H. Kauffman and D. Radford. Bi-oriented quantum algebras, and a generalized Alexander polynomial for virtual links. *Contemp. Math.* **318** (2003) 113-140.

[12] T. Kishino and S. Satoh. A note on non-classical virtual knots. *J. Knot Theory Ramifications* **13** (2004) 845-856.

[13] D. Lam and S. Nelson. Classsification of finite Alexander biquandles. Preprint available at arXiv.org:math.QA/0611887

[14] S. Nelson and J. Vo. Matrices and finite biquandles. *Homology, Homotopy and Applications* **8** (2006) 51-73.